BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama program magang di PT. Satya Solusindo Indonesia, penulis ditempatkan sebagai *automation engineer*. Tanggung jawab utama difokuskan pada dukungan teknis bagi para *software* dan *automation engineer* dalam merampungkan sejumlah proyek terkait hardware, antara lain penentuan kriteria spesifikasi peralatan atau mesin, penyusunan desain 3D, serta pembuatan flowchart, gambaran sistem, dan dokumen teknis.

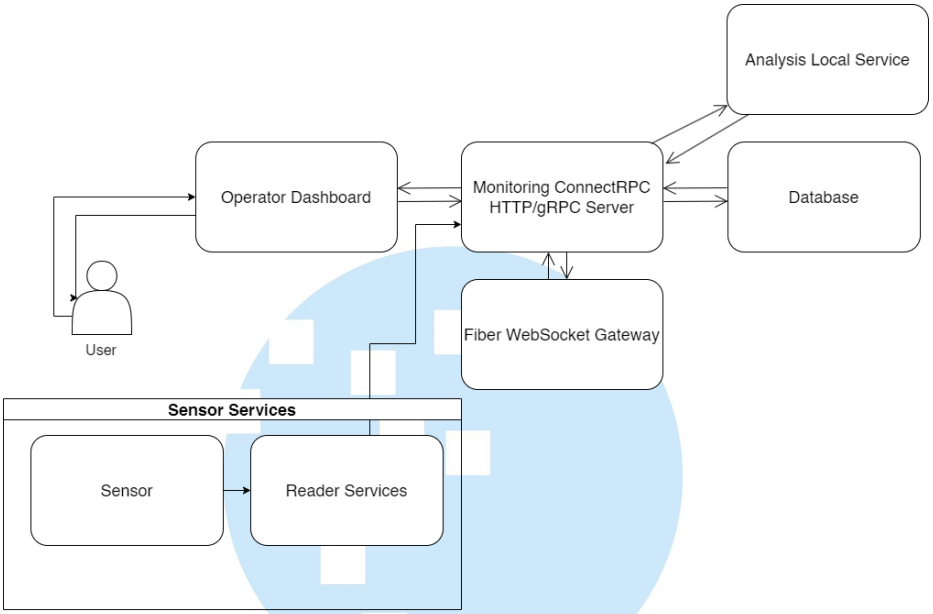
Implementasi sistem pemantauan *reciprocating compressor* berbasis sensor vibrasi dipercayakan kepada penulis selama program magang. Tanggung jawab ini meliputi perancangan desain casing komponen inti sistem *monitoring*, pengembangan *website*, hingga analisis data sensor berdasarkan standar ISO 10816 sebagai patokan evaluasi getaran mesin *reciprocating compressor*. Alur koordinasi dilakukan dengan interaksi tatap muka kasual dengan *supervisor* yang bertujuan untuk memecahkan hambatan minor secara instan.

3.2 Tugas dan Uraian Kerja Magang

Selama program magang, kontribusi diberikan pada proyek sistem pemantauan sensor getaran pada mesin kompresor berdasarkan standar ISO 10816 secara waktu nyata (*real-time*). Standar ini digunakan untuk menentukan apakah nilai getaran berada dalam kondisi normal atau memerlukan perhatian lebih lanjut. Secara garis besar, sistem pemantauan ini terdiri atas sensor getaran, sebuah Raspberry Pi 4 yang berfungsi sebagai server *monitoring*, serta sebuah *website* yang digunakan sebagai *dashboard* interaksi antara mesin dan pengguna. *Server monitoring* dibagi menjadi dua bagian, yaitu *frontend* dan *backend*.

Bagian *backend* bertanggung jawab untuk membaca nilai sensor, mengirimkan data ke *database*, serta melakukan analisis berdasarkan hasil pembacaan sensor. Sementara itu, bagian *frontend* menyajikan data dan hasil analisis kepada pengguna melalui *dashboard*.

Struktur dan hubungan antar komponen dalam sistem pemantauan sensor getaran pada *reciprocating compressor* ditunjukkan pada Gambar 3.1, yang merupakan representasi dari topologi sistem secara keseluruhan.

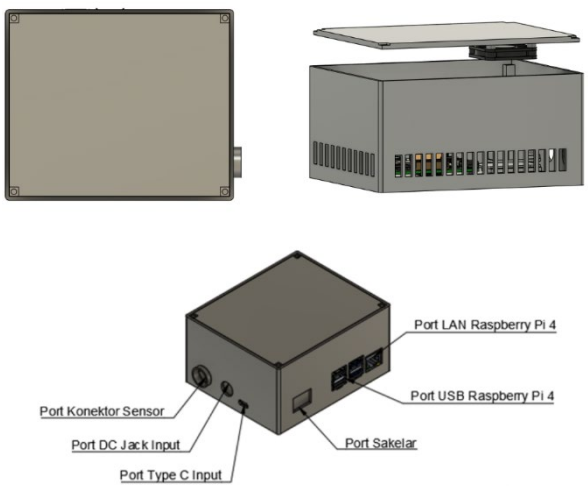


Gambar 3.1 Topologi Sistem Monitoring Getaran pada *Reciprocating Compressor*

Gambaran topologi sistem pada Gambar 3.1 menjadi dasar bagi pembahasan terperinci pada bagian selanjutnya, yang meliputi arsitektur perangkat keras, perangkat lunak, serta mekanisme komunikasi dan pertukaran data antar komponen dalam sistem *monitoring*.

3.2.1. Desain *Casing* dan Tata Letak Fisik

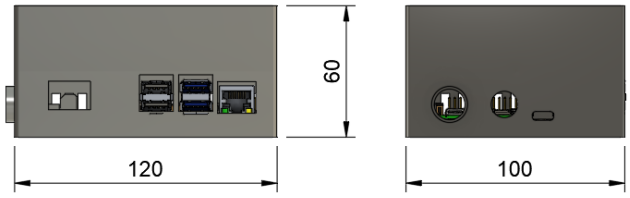
3.2.1.1. Desain Casing Sistem



Gambar 3.2 Konsep Desain Produk

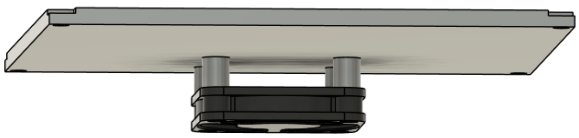
Desain *casing* berfungsi melindungi perangkat keras dan menjaga suhu komponen agar tetap stabil. *Casing* dirancang berukuran kompak sehingga mudah dibawa dan dipasang. Gambar 3.2 menunjukkan desain *casing* sistem dalam bentuk model 3D yang dibuat menggunakan aplikasi Autodesk Fusion.

Desain *casing* sistem *monitoring* didasarkan pada tiga tujuan (*goals*) utama yang disesuaikan dengan kebutuhan klien. Pertama, perancangan harus memenuhi persyaratan ukuran ringkas (*compact*) dan sekecil mungkin. Oleh karena itu, *casing* memiliki dimensi 12 cm × 10 × cm × 6 cm, yang ditunjukkan pada Gambar 3.3.



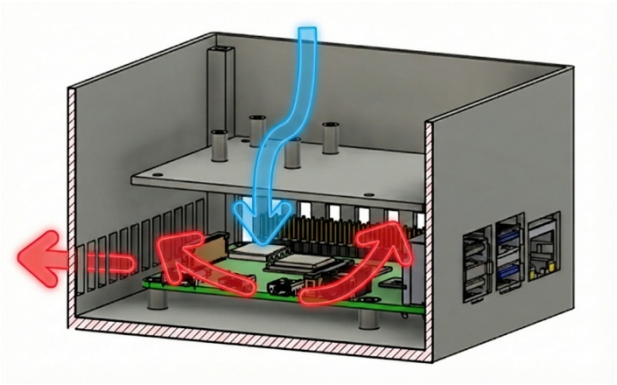
Gambar 3.3 Dimensi *Casing* Sistem Monitoring

Tujuan kedua, *casing* harus mendukung sistem pendinginan yang efektif. Hal ini krusial mengingat lingkungan (*environment*) pabrik industri klien diklaim bersuhu panas. Untuk mengatasi hal tersebut, lubang sirkulasi udara didesain pada bagian samping kiri dan belakang untuk memastikan pembuangan udara panas. Sistem pendinginan dilakukan dengan meletakkan kipas berukuran 4 cm × 4 cm dengan spesifikasi: tegangan operasional 4,5-5 V, arus 0,2 A, dan kecepatan 13.200 RPM (*Revolutions Per Minute*). Kipas pendingin diposisikan pada bagian penutup komponen dan dikencangkan menggunakan baut berukuran M3.



Gambar 3.4 *Desain Penutup Casing Komponen*

Penggunaan kipas direkomendasikan karena tanpa kipas, suhu Raspberry Pi 4 meningkat secara signifikan selama pengujian beban[1]. Kipas dipasang pada mounting bagian penutup *casing* dengan udara ditiupkan ke arah dalam. Visualisasi sirkulasi udara yang terjadi dapat dilihat pada Gambar 3.4.



Gambar 3.5 Visualisasi Sirkulasi Udara Panas dan Segar yang terjadi pada *Casing*

Visualisasi udara panas pada Gambar 3.4 direpresentasikan dengan panah merah, sedangkan udara segar dari lingkungan direpresentasikan dengan panah biru. Setelah komponen ditempatkan, *casing* dilakukan pengujian pengukuran suhu . Pengujian dilakukan di kantor dengan suhu udara 25ºC. Berdasarkan Tabel 3.1, hasil pengujian menunjukkan bahwa bila Raspberry Pi 4 dinyalakan tanpa kipas, suhu dapat mencapai 55ºC. Namun, ketika kipas dipasang, suhu operasional ketika baru dinyalakan mencapai 40ºC dan stabil di 41ºC.

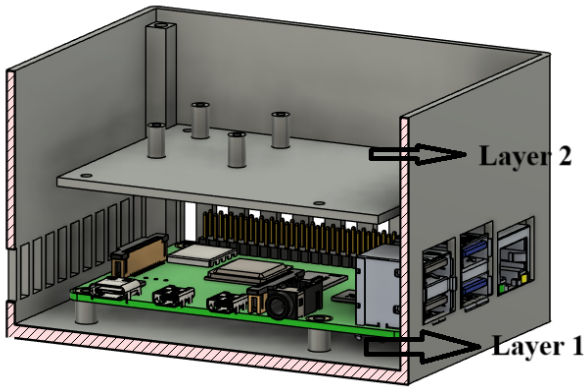
Raspberry Pi 4 Model B juga dilakukan uji beban tinggi (*stress test*) untuk mengetahui suhu operasional ketika beban tinggi (*high load*) dengan dua kali percobaan, yaitu durasi 1 jam (percobaan pertama) dan 5 menit (percobaan kedua). Pengujian dilakukan menggunakan *dietpi-config_stress.log*. Pengujian dilakukan sebanyak dua kali, yaitu percobaan pertama berdurasi 1 jam dan percobaan kedua berdurasi 5 menit. Dalam salah satu percobaan (Percobaan Kedua), Raspberry Pi 4 mampu mendapatkan suhu minimal 46ºC dan suhu maksimal 64ºC. Berdasarkan hasil keseluruhan pengujian, terbukti bahwa desain *casing* yang diterapkan mampu menjaga mikroprosesor

Raspberry Pi 4 Model B agar tetap mencapai suhu operasional yang aman.

Tujuan ketiga, desain harus menjamin kemudahan perawatan (*maintenance*) komponen *hardware*. Untuk mengantisipasi kebutuhan kalibrasi atau atur ulang (*reset*) perangkat *monitoring*, *casing* dilengkapi dengan lubang port USB dan LAN untuk Raspberry Pi 4.

3.2.1.2. Tata Letak Komponen di Dalam *Casing*

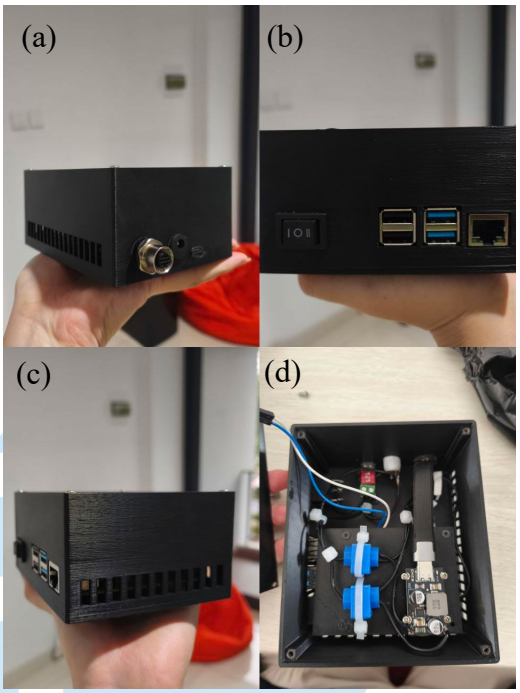
Tata letak komponen akan dibagi menjadi dua lapisan (*layer*), yaitu lapisan pertama dan lapisan kedua. Pembagian lapisan ini bertujuan untuk menjaga ukuran sistem yang dirancang agar se-ringkas (*compact*) mungkin. Lapisan pertama berada di bagian dasar *casing*, sementara lapisan kedua diletakkan di atas CAN HAT RS-485.



Gambar 3.6 Layer pada *Casing* Komponen

Lapisan pertama mencakup komponen seperti konektor M12, modul USB-C PD Decoy, soket masukan DC (DC *Jack input*), sakelar tiga arah, dan Raspberry Pi 4 Model B. Sementara itu, lapisan kedua (layer 2) ditempati oleh modul *step-down* USB-C.

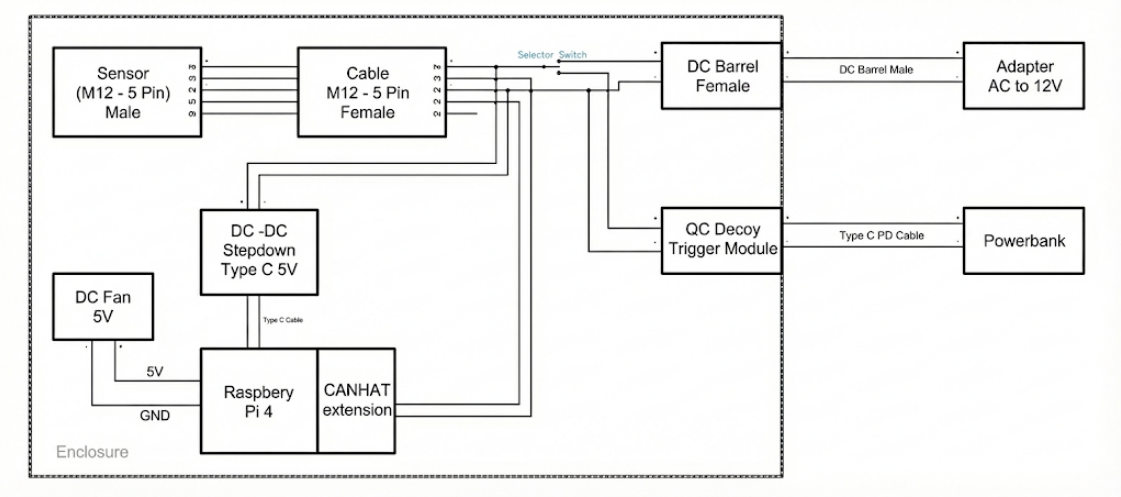
3.2.1.3. Realisasi Desain



Gambar 3.7 Realisasi Fisik Komponen Sistem: (a) tampak isometrik; (b) tampak kanan; (c) tampak belakang; (d) tata letak komponen internal sistem.

Realisasi fisik dari *casing* sistem *monitoring* getaran dicetak menggunakan printer 3D dengan *infill* sebesar 60% dan material filamen ABS. Proses pencetakan ini menghasilkan bentuk yang sesuai dengan rancangan desain 3D yang telah dibuat sebelumnya. Material ABS dipilih karena memiliki kekuatan mekanis yang baik serta ketahanan termal yang tinggi, sehingga mampu melindungi komponen elektronik ketika sistem beroperasi pada suhu yang relatif tinggi. *Casing* yang telah dicetak kemudian dirakit bersama Raspberry Pi dan modul pendukung lainnya, sehingga berfungsi sebagai wadah utama yang melindungi dan menyatukan seluruh komponen sistem *monitoring*.

3.2.2. Koneksi dan Skematik Elektrikal Sistem



Gambar 3.8 Skematik Elektrikal Sistem

Sistem *monitoring* memiliki dua opsi sumber daya utama, yaitu menggunakan adaptor AC atau *powerbank*. Adaptor yang digunakan memiliki spesifikasi minimal 12 volt dengan arus sebesar 0,5 ampere. Sementara itu, jika menggunakan powerbank, perangkat harus mendukung teknologi *Power Delivery* (PD) karena port USB Type-C atau micro-USB konvensional umumnya hanya mampu menyediakan daya sebesar 10 watt. Berdasarkan hasil pengukuran yang dilakukan oleh penulis, terdapat dua konfigurasi suplai daya yang dapat digunakan dari powerbank, yaitu 12,1 volt dengan arus 0,5 ampere, atau 20,1 volt dengan arus 0,35 ampere. Pemilihan sumber daya dilakukan melalui *switch* tiga arah yang mengatur jalur masukan (*input*) sesuai kebutuhan.

Setelah sumber daya dipilih, tegangan masukan akan dibagi menjadi dua jalur utama: satu untuk sensor dan satu untuk Raspberry Pi. Berdasarkan spesifikasi teknis, Raspberry Pi 4 hanya dapat menerima tegangan input sebesar 5 VDC. Oleh karena itu, digunakan modul step-down DC-DC Type-C untuk menurunkan tegangan dari powerbank atau adaptor agar sesuai dengan kebutuhan Raspberry Pi dan mencegah terjadinya *overvoltage*.

3.2.3. Komponen *Hardware*

Bagian ini menjelaskan komponen perangkat keras yang digunakan dalam sistem *monitoring* getaran. Setiap komponen memiliki peran spesifik dalam proses akuisisi data, pemrosesan awal, dan pengiriman informasi ke server *monitoring*.

3.2.3.1. Sensor *Vibration* (Getar)

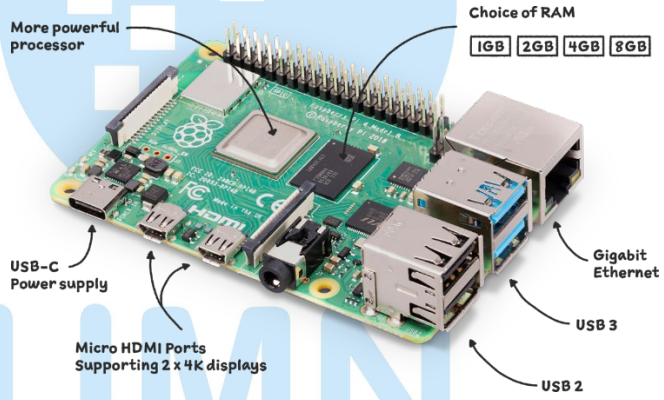
Gambar 3.9 *Industrial Vibration and Temperature Sensor* QM30VT2
Sumber: Banner (2025)

Sensor getar QM30VT2 digunakan pada sistem ini untuk mendeteksi suhu dan memonitor kesehatan mesin berdasarkan getaran yang terdeteksi. Berdasarkan *datasheet* komponen, sensor QM30VT2 beroperasi pada tegangan 10 hingga 30 VDC dengan laju baud (*baud rate*) 19200 bps. Selain itu, sensor ini diklaim mampu memberikan keluaran (*output*) berupa Kecepatan RMS (*RMS Velocity*), Percepatan Frekuensi Tinggi RMS (*RMS High-Frequency Acceleration*), Kecepatan Puncak (*Peak Velocity*), serta masukan (*input*) lain yang telah dipraproses (*pre-processed*) dari bentuk gelombang getaran (*vibration waveforms*) [2]. Sensor menggunakan standar komunikasi industri RS-485 untuk *output* data.

Sensor QM30VT2 mengeluarkan dua puluh dua (22) jenis data yang tersimpan pada alamat *register* yang berbeda.

Fitur ini memberikan fleksibilitas bagi pengguna untuk memilih variabel data yang ingin ditampilkan atau diolah. Dalam analisis sistem *monitoring* getaran (*vibration monitoring*), Kecepatan RMS Sumbu Z (*Z-Axis RMS Velocity*) dan Kecepatan RMS Sumbu X (*X-Axis RMS Velocity*), keduanya dalam satuan mm/detik, digunakan sebagai variabel analisis utama apakah telah sesuai dengan standar ISO 10816. Pengukuran suhu dan kecepatan menjadikan sensor QM30VT2 ideal untuk aplikasi dasbor (*dashboard*) dan sistem *monitoring* getaran (*vibration monitoring*).

3.2.3.2. *Microprocessor* (Raspberry Pi 4 Model B)



Gambar 3.10 Raspberry Pi 4 Model B
Sumber: Raspberry Pi (2025)

Raspberry Pi 4 Model B merupakan mikrokomputer papan tunggal (*Single-Board Computer* atau SBC) dengan spesifikasi yang mencakup RAM 4 GB dan prosesor Broadcom BCM2711, yaitu *quad-core Cortex*-A72 (ARM v8) 64-bit SoC berkecepatan 1,8 GHz[3]. Selain itu, perangkat ini memiliki 40 pin GPIO (*General Purpose Input/Output*) yang beroperasi pada tegangan 3,3 volt. Secara bawaan, komponen ini tidak mendukung protokol komunikasi RS-485. Oleh karena itu, RS-485 CAN HAT digunakan untuk mengkonversi protokol

komunikasi dari RS-485 ke UART (*Universal Asynchronous Receiver/Transmitter*). Konversi ini diperlukan karena Raspberry Pi 4 Model B hanya mendukung protokol komunikasi serial seperti UART dan SPI. Aspek penting lainnya yang perlu diperhatikan adalah kebutuhan catu daya, di mana Raspberry Pi 4 Model B memerlukan daya masukan sebesar 5V/3A DC.

3.2.3.3. Modul Pendukung

Modul pendukung merupakan perangkat tambahan yang berfungsi memastikan proses akuisisi data dan integrasi sensor dengan Raspberry Pi dapat berjalan secara stabil. Modul-modul ini tidak menghasilkan data secara langsung, tetapi memiliki peran teknis penting dalam penyediaan jalur komunikasi dan kestabilan suplai daya sehingga sistem *monitoring* dapat beroperasi sesuai kebutuhan.

1) CAN HAT RS-485

Gambar 3.11 CAN HAT RS-485

Perangkat RS-485 CAN HAT digunakan sebagai modul komunikasi tambahan untuk melakukan konversi protokol dari RS-485 ke UART (Universal Asynchronous *Receiver/Transmitter*). Konversi ini diperlukan karena protokol komunikasi yang digunakan oleh sensor getar adalah RS-485, yang merupakan standar dalam lingkungan industri. RS-485 sendiri adalah metode komunikasi data serial yang mampu melakukan transmisi data hingga jarak 1,2 kilometer, dan dikembangkan pada tahun 1983[4].

2) Modul Step-Down DC-DC Type-C

Gambar 3.12 Module Step Down DC-DC Type-C

Modul *step-down* berfungsi sebagai regulator daya yang menurunkan tegangan masukan eksternal menjadi 5 volt, yaitu tegangan operasional yang dibutuhkan oleh Raspberry Pi 4. Penurunan tegangan ini diperlukan karena sumber daya eksternal yang digunakan memiliki keluaran minimal 12 volt hingga maksimum 20 volt, bergantung pada protokol *Power Delivery* (PD) yang dinegosiasikan melalui modul *USB-C PD decoy*. Dengan adanya modul *step-down*, suplai daya ke Raspberry Pi dapat tetap stabil dan sesuai spesifikasi.

3) USB Power Delivery (PD) *Decoy*

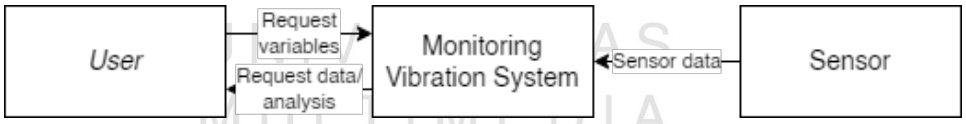
Gambar 3.13 USB PD *Decoy*

Modul USB-C PD *Decoy* merupakan modul elektronik yang berfungsi untuk menginstruksikan adaptor daya atau pengontrol USB *Type*-C agar mengeluarkan

tegangan yang lebih tinggi dari standar 5V. Penggunaan modul ini sangat penting bagi sistem untuk memastikan komponen *monitoring* mendapatkan suplai daya yang sesuai, khususnya untuk memenuhi spesifikasi tegangan minimum sensor getar sebesar 12 V. Modul ini mendukung variasi keluaran tegangan 9 V, 12 V, 15 V, dan 20 V.

3.2.4. Struktur Komunikasi Sistem Monitoring

Pada sistem *monitoring* getaran, Raspberry Pi 4 berperan sebagai server sekaligus *access point* yang membentuk jaringan lokal (*local hotspot*). Raspberry Pi secara otomatis menghasilkan SSID dan alamat IP lokal yang digunakan sebagai jalur komunikasi internal. Perangkat *client*—seperti laptop atau *smartphone*—dapat terhubung ke *hotspot* tersebut tanpa memerlukan koneksi internet eksternal. Setelah terhubung, pertukaran data pembacaan sensor antara *client* dan Raspberry Pi berlangsung melalui alamat IP lokal yang diberikan oleh Raspberry Pi. Dengan mekanisme ini, client dapat mengakses *dashboard monitoring* melalui browser dan melakukan permintaan variabel/analisis data (*request*) secara langsung ke server yang berjalan di Raspberry Pi. Arsitektur ini memastikan sistem tetap dapat beroperasi secara mandiri (*stand-alone*) di lingkungan industri tanpa ketergantungan pada infrastruktur jaringan eksternal.

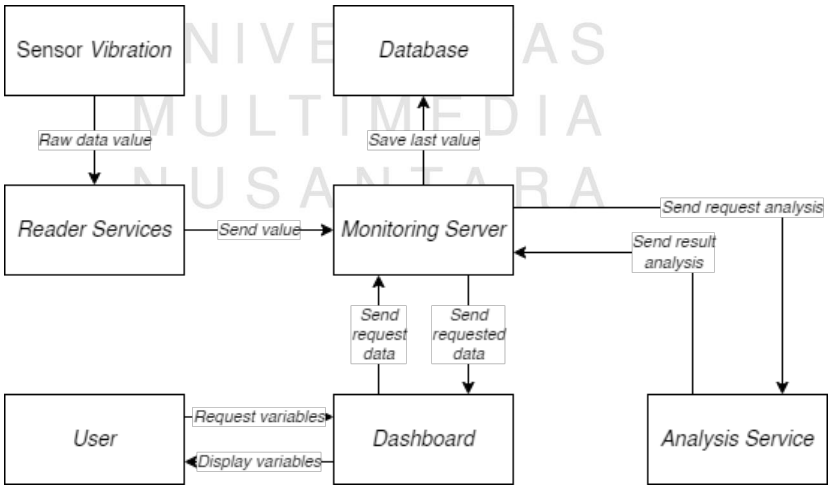


Gambar 3.14 DFD Level 0 - *Main System*

Sistem *monitoring* akan dimulai pada saat sensor telah terpasang dan dapat mengirimkan data dari sensor ke *dashboard reader services*. *Reader service* adalah daemon pembaca sensor yang mana memuat konfigurasi dan menginisialisasi koneksi Modbus RTU serta *database*, menjalankan *loop* terus-menerus untuk mengambil daftar *metric* dari service, dan membaca nilai data register sensor lewat Modbus tiap sekitar 3 detik dengan deklarasi

baud rate 19200. Data mentah dari sensor dibaca melalui protokol Modbus RTU, kemudian dikonversi menjadi nilai numerik dengan asumsi urutan *byte* (*byte order*) big-*endian*. Nilai numerik mentah tersebut selanjutnya diskalakan sesuai dengan nilai *metric.Scale* (dengan rumus: nilai = *mentah / metric.Scale*) untuk memperoleh nilai akhir yang diproses. Nilai *realtime* dikirim ke bagian *backend* menggunakan RPC berbasis HTTP melalui pustaka ConnectRPC (*http.Client*). Alamat *server default* adalah http://192.168.1.50:9000, namun alamat tersebut dapat diubah melalui variabel lingkungan (*environment variable*) GRPC_SERVER.

Setelah data pembacaan sensor terkirim, pengguna (*user*) dapat melihat nilai tersebut melalui dasbor (*dashboard*) dengan menghubungkan perangkat (*device*) yang digunakan untuk mengakses situs web *monitoring* (sebagai *client*). Pengguna dapat memilih variabel yang ingin ditampilkan pada bagian pengaturan (*setting*), yaitu pada konfigurasi sensor (*sensor configuration*). Pada dasbor, pengguna juga dapat mengajukan permintaan untuk melakukan analisis melalui fitur perekaman data (*record data*). Komunikasi antara fitur situs web dan klien (*client*) dilakukan menggunakan teknologi *WebSocket* dengan menyediakan titik akhir (*endpoint*) yang sama agar dapat terhubung. Penjelasan alur data (*data flow*) lebih lanjut dapat dilihat pada Gambar 3.14.

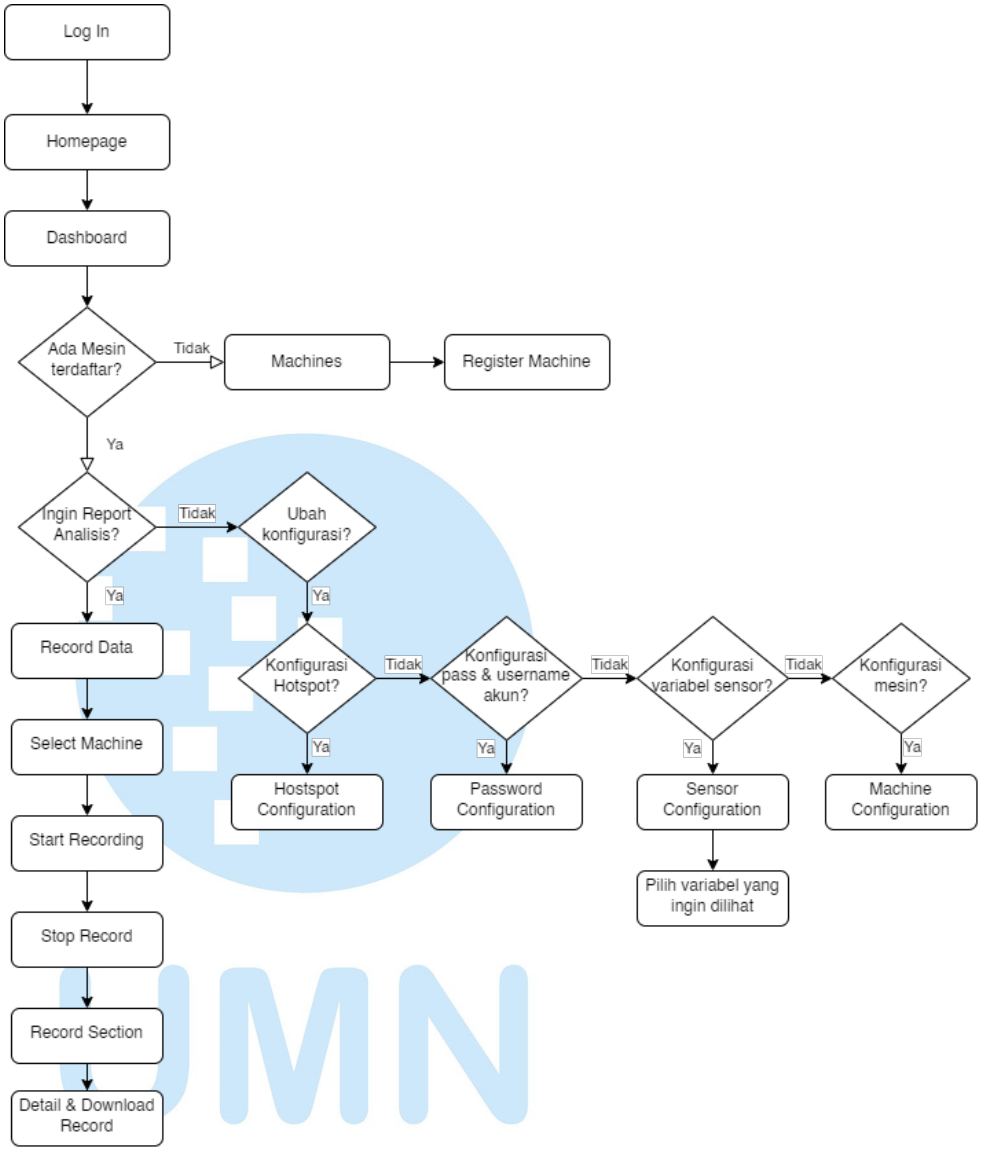


Gambar 3.15 DFD Sistem-DFD Level 1 - *Monitoring Vibration System*

3.2.5. Komponen *Software*

3.2.4.1. *Frontend*

Frontend pada sistem *monitoring* getaran berfungsi sebagai antarmuka utama yang digunakan oleh pengguna untuk mengakses informasi kondisi mesin, melakukan konfigurasi, serta melihat hasil analisis data. Antarmuka ini disajikan dalam bentuk *dashboard* berbasis *web* yang dapat diakses melalui perangkat apa pun yang terhubung ke *hotspot* lokal yang dibentuk oleh Raspberry Pi 4. Dengan mekanisme ini, seluruh proses interaksi antara pengguna dan sistem berlangsung melalui jaringan lokal menggunakan alamat IP Raspberry Pi, sehingga *dashboard* dapat diakses secara mandiri tanpa memerlukan koneksi internet eksternal. *Frontend* dirancang untuk menampilkan data secara *real-time*, menyediakan navigasi yang intuitif, serta mendukung berbagai fungsi operasional seperti pemilihan mesin, perekaman data, dan pengaturan variabel sensor. Alur proses pengguna berinteraksi dengan *dashboard monitoring* divisualisasikan dalam bentuk *flowchart* pada Gambar 3.15. *Flowchart* ini menggambarkan langkah-langkah yang dilakukan pengguna mulai dari *proses login*, navigasi *dashboard*, pemilihan mesin, konfigurasi sistem, hingga perekaman dan pengunduhan data.



Gambar 3.16 *Flowchart User Flow*

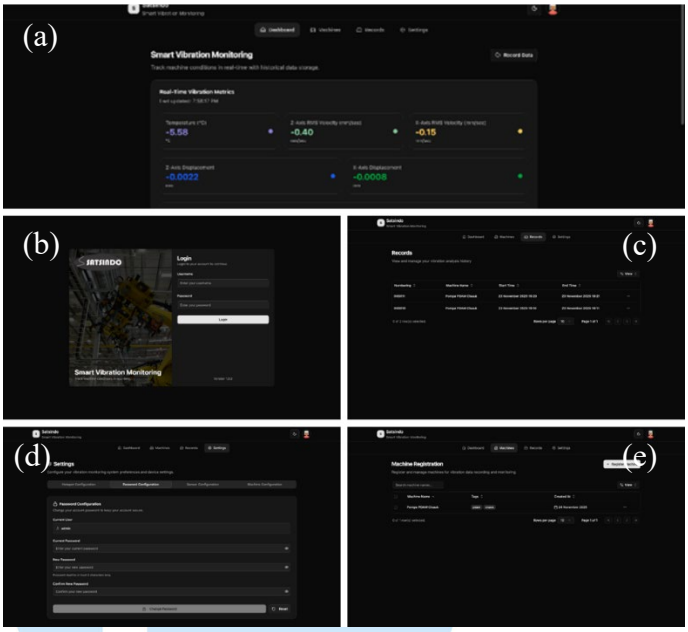
Berdasarkan Gambar 3.15, *frontend* menyediakan beberapa jalur interaksi yang fleksibel sesuai kebutuhan pengguna. Setelah berhasil masuk ke sistem, pengguna diarahkan ke *dashboard* untuk melihat status mesin. Jika belum ada mesin yang terdaftar, *frontend* menampilkan opsi untuk melakukan registrasi mesin. Selain itu, pengguna dapat mengakses menu konfigurasi seperti pengaturan *hotspot*, akun, variabel sensor, dan mesin. Proses perekaman data *frontend*

menyediakan antarmuka untuk memilih mesin, memulai perekaman, menghentikan perekaman, serta melihat dan mengunduh hasil rekaman. Seluruh interaksi ini dilakukan melalui permintaan ke backend melalui *WebSocket*.

Perancangan aplikasi *website* menggunakan kerangka *Nexus Dashboard*, dengan sisi *frontend* dikembangkan menggunakan React dan React Router untuk menangani proses *routing* antar halaman. Pada Halaman Utama (*Main Page*), antarmuka *frontend* dibagi menjadi empat subbagian utama, yaitu *Dashboard*, *Machine*, *Record*, dan *Setting*. Subbagian *Dashboard* berfungsi menampilkan ringkasan status mesin, tren grafik getaran sensor secara waktu nyata (*real-time*), visualisasi grafik *time-series*, serta indikator status kesehatan mesin. Subbagian *Machine* digunakan untuk memilih mesin yang sedang di*monitoring* atau mendefinisikan mesin kompresor baru yang akan dimasukkan ke dalam sistem. Subbagian *Record* menyediakan akses terhadap hasil analisis perekaman data yang diperoleh setelah pengguna (*user*) melakukan proses *record data* melalui *Dashboard*. Sementara itu, subbagian *Setting* memungkinkan pengguna melakukan pengaturan sistem seperti mengubah sandi *hotspot*, melakukan *shutdown* atau *restart* perangkat Raspberry Pi, serta konfigurasi dasar lainnya yang berkaitan dengan perangkat keras.

Frontend tidak hanya menampilkan struktur utama pada Halaman Utama (*Main Page*), tetapi juga menyediakan halaman detail mesin dengan visualisasi lebih mendalam seperti grafik *time-series*, indikator alarm, dan histori data. Selain itu, tersedia halaman login untuk autentikasi guna memastikan keamanan akses, serta fitur tambahan seperti pencarian atau penyaringan data (*filtering*) dan ekspor atau unduh laporan untuk mendukung

analisis dan dokumentasi. Visualisasi *dasboard* sistem *monitoring* dapat dilihat pada Gambar 3.16.



Gambar 3.17 Visualisasi *Dashboard* Sistem *Monitoring*: (a) *main page dashboard website*; (b) *login page website*; (c) *main page* bagian *record website*; (d) *main page* bagian *setting website*; (e) *main page* bagian *machine*

Proses *build* dan pengembangan dilakukan menggunakan *Vite*. Untuk desain antarmuka pengguna (*User Interface* atau UI), digunakan sejumlah pustaka (*library*) modern, yaitu Tailwind CSS sebagai utilitas CSS utama, Radix UI dan shadcn/ui untuk penyediaan komponen UI modular yang mudah dikustomisasi, serta Lucide React untuk tampilan ikon yang konsisten dan modern. Kombinasi teknologi ini memastikan tampilan dasbor tetap bersih, responsif, dan mudah digunakan baik pada perangkat *desktop* maupun *mobile*.

Frontend berkomunikasi dengan *backend* melalui konsumsi data dari API (*Application Programming Interface*), baik menggunakan pola RESTful API maupun komunikasi gRPC di atas protokol HTTP, sesuai dengan kebutuhan integrasi sistem. Data getaran, status mesin, dan hasil analisis di-*fetch*

secara berkala, dilengkapi dengan mekanisme penanganan *loading, success*, dan *error* untuk memastikan pengalaman pengguna (*User Experience* atau UX) tetap stabil dan informatif. Setiap halaman dirancang responsif dan mengedepankan UX yang intuitif, dengan navigasi yang jelas dan komponen UI (*User Interface*) yang konsisten. Dengan integrasi teknologi modern tersebut, *frontend* aplikasi ini mendukung tim perawatan (*maintenance*) dalam melakukan *monitoring* kondisi mesin secara efisien, memahami pola getaran melalui visualisasi yang informatif, serta memungkinkan pengambilan keputusan cepat berbasis data yang tersaji secara akurat.

3.2.4.2. Backend

Backend pada sistem *monitoring* getaran dibangun menggunakan bahasa pemrograman Go. *Backend* ini bertugas mengelola pertukaran data antara *frontend* aplikasi dan subsistem atau modul lain, salah satunya aplikasi analisis lokal yang terpisah. Dengan arsitektur berbasis RESTful API dan gRPC, *backend* menyediakan titik akhir (*endpoint*) bagi *frontend* untuk melakukan operasi seperti mengambil data getaran mesin, menyimpan hasil inspeksi, serta mendistribusikan informasi ke berbagai bagian halaman aplikasi secara waktu nyata (*real-time*).

Selain pengelolaan *service* API, *backend* ini memiliki fitur utama berupa fungsi *generate* berkas Excel secara otomatis dari data hasil akuisisi getaran yang diterima. Data yang telah dikumpulkan oleh *backend* selanjutnya dieksport dalam format Excel (.xlsx) melalui titik akhir (*endpoint*) khusus, sehingga pengguna (*user*) atau sistem dapat mengunduh berkas (*file*) tersebut untuk dilakukan analisis lanjut secara lokal. Dengan demikian, *backend* ini tidak menangani proses analisis getaran secara mendalam, namun lebih berperan dalam orkestrasi data,

otomasi pelaporan data mentah, serta memastikan integrasi yang lancar antara proses akuisisi, penyimpanan, hingga distribusi data ke aplikasi lain.

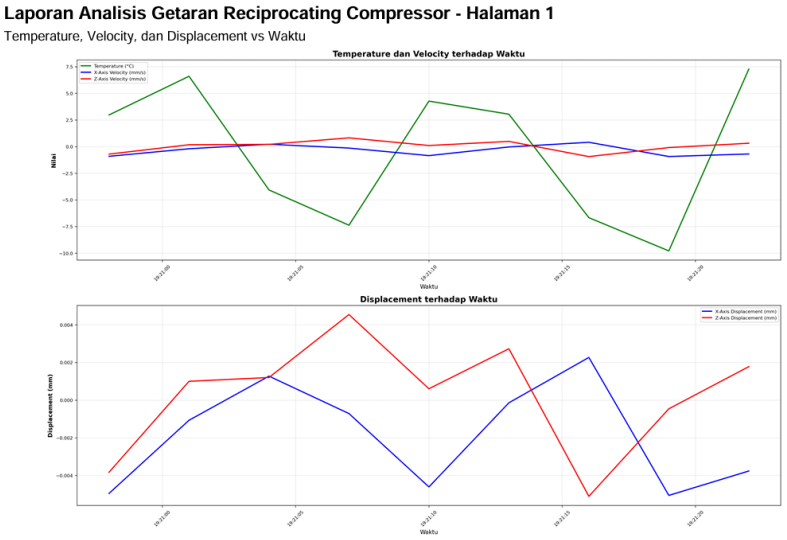
3.2.4.3. Program *Report Analysis*

Report Analisis dilakukan secara lokal yang mana dimulai dengan menerima berkas Excel (*file Excel*) yang berisi data timestamp dan nilai getaran per sumbu. Program kemudian membaca semua sheet, melakukan pembersihan dan penstandardisasian nama kolom, lalu menggabungkan data berdasarkan kolom Timestamp untuk membentuk satu DataFrame. Dari sana, alur berlanjut ke fungsi analisis: untuk setiap baris, program memeriksa kolom Kecepatan RMS (RMS velocity) yaitu x_axis_rms_velocity_mm_per_sec dan z_axis_rms_velocity_mm_per_sec—dan mengklasifikasikan setiap nilai sumbu menggunakan fungsi classify_vibration_general (mengacu pada kelas mesin I–IV). Selanjutnya, program menghitung resultan kecepatan (v) = untuk mendapatkan satu indikator total getaran, lalu mengklasifikasikan resultan tersebut.

Klasifikasi zona mengikuti standar ISO 10816, yaitu A (baik), B (cukup), C (tidak disarankan), dan D (tidak dapat diterima), dengan ambang batas khusus per kelas mesin sebagai berikut: Kelas I:A≤ 0,71; B ≤ 1,8; C ≤ 4,5 (D > 4,5); Kelas II: A ≤ 1,12; B ≤ 2,8; C ≤ 7,1 (D > 7,1); Kelas III: A ≤ 1,8; B ≤ 4,5; C ≤ 11,2 (D > 11,2); Kelas IV: A ≤ 2,8; B ≤ 7,1; C ≤ 18,0 (D > 18,0)[5]. Keputusan akhir untuk setiap waktu diambil dari kondisi terburuk (*worst-case*) antara sumbu X, sumbu Z, dan resultan.

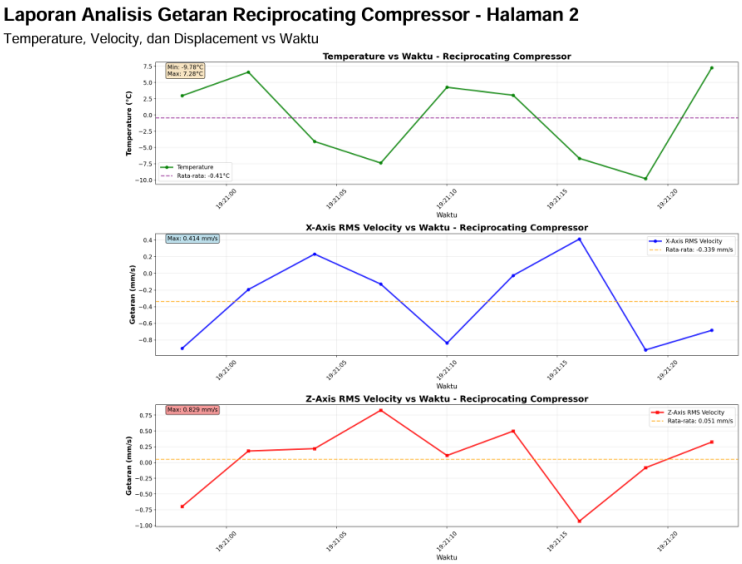
Berkas hasil analisis getaran kompresor torak (*reciprocating compressor*) disimpan dalam format PDF berisi 4 halaman dengan berbagai visualisasi untuk mendukung

interpretasi data. Laporan analisis ini disajikan dalam empat halaman yang masing-masing menampilkan aspek berbeda dari kondisi getaran mesin. Visualisasi data disajikan menggunakan beberapa jenis grafik yang dirancang untuk memudahkan pemahaman terhadap performa dan kondisi kompresor.



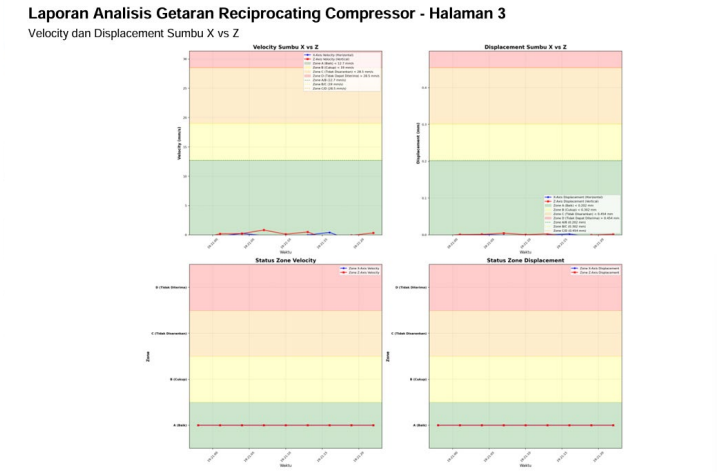
Gambar 3.18 Analisis Perubahan Resultan Kecepatan Getaran dan Suhu terhadap Waktu

Gambar 3.18 menampilkan analisis grafik *line plot* mengenai perubahan temperatur dan *velocity* terhadap waktu, serta *displacement* pada sumbu X dan Z terhadap waktu. Garis hijau menggambarkan perubahan nilai temperatur terhadap waktu, garis merah menunjukkan perubahan *displacement* sumbu-X terhadap waktu, dan garis biru menunjukkan perubahan *displacement* sumbu-Z terhadap waktu. Grafik-grafik ini memberikan gambaran mengenai tren perubahan parameter operasional kompresor dalam periode waktu tertentu.



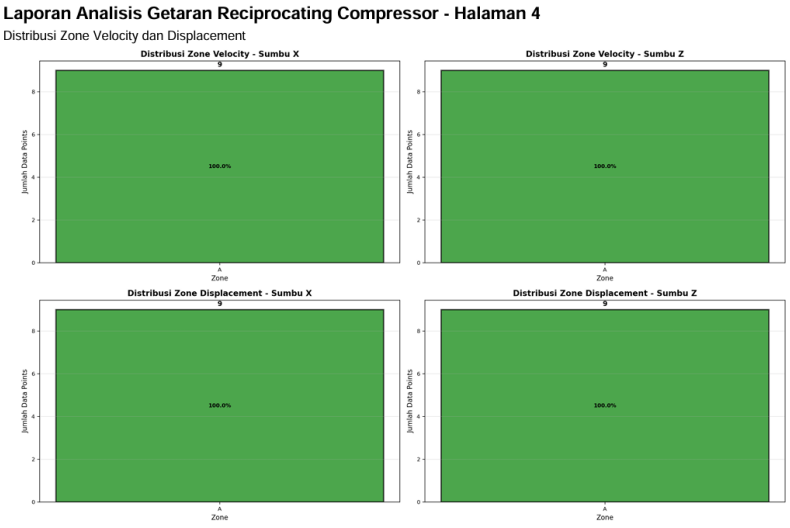
Gambar 3.19 Grafik Temperatur, Kecepatan Axis-X dan Axis-Z terhadap Waktu

Grafik line plot pada Gambar 3.18 memerinci perubahan temperatur terhadap waktu dalam zona ambang batas tertentu. Data tersebut disandingkan dengan parameter RMS *velocity* sumbu X dan sumbu Z, serta nilai rerata dari seluruh hasil perekaman data.



Gambar 3.20 Analisis Grafik Kecepatan dan *Displacement* Sumbu X dan Z, zona waktu terhadap waktu

Gambar 3.19 menyajikan analisis yang berbeda, yaitu grafik sebar (*scatter plot*) yang menampilkan hubungan antara *velocity* dan *displacement* pada sumbu X versus sumbu Z. Berbeda dengan visualisasi sebelumnya, gambar ini memiliki latar belakang zona berwarna yang merepresentasikan klasifikasi kondisi mesin (Zone A/Baik, Zone B/Cukup, Zone C/Tidak Direkomendasikan, dan Zone D/Bahaya). Di bawahnya, terdapat grafik status zona yang menunjukkan klasifikasi kondisi *velocity* dan *displacement* berdasarkan zona-zona tersebut sepanjang waktu pengukuran. Grafik ini berfungsi untuk memantau apakah kondisi getaran kompresor berada pada zona aman atau tidak.



Gambar 3. 21 Distribusi Zona Kecepatan dan *Displacement*

Terakhir, halaman keempat menampilkan grafik batang (*bar chart*) yang menunjukkan distribusi persentase data pada masing-masing zona untuk *velocity* dan *displacement* pada sumbu X dan Z. Berdasarkan data yang ada, seluruh data (100%) berada pada zona tertentu, yang mengindikasikan konsistensi kondisi operasional kompresor selama periode pengukuran.

Kombinasi berbagai jenis grafik ini—mulai dari *line plot*, *line plot* dengan zona, *scatter plot*, hingga *bar chart*—memberikan analisis yang komprehensif dan multi-dimensi terhadap kondisi getaran kompresor torak, memungkinkan interpretasi yang lebih akurat untuk keperluan *maintenance* dan *monitoring* kondisi mesin. Hasil analisis ini kemudian dapat dilihat pada subbagian Record di dasbor aplikasi.

3.3 Kendala yang Ditemukan

Permasalahan utama yang dihadapi dalam pengembangan sistem *monitoring* ini meliputi kendala teknis dan logistik yang secara kolektif memperlambat progres proyek. Di sisi pengembangan perangkat lunak (*software*), kebutuhan untuk menguasai *tools* pembuatan situs web (*website*) modern seperti React dan Tailwind menjadi suatu keharusan, karena kurangnya keahlian di bidang tersebut menghambat efisiensi pengembangan antarmuka pengguna (*User Interface* atau UI). Sementara itu, di sisi manufaktur dan perakitan fisik, ditemui beberapa hambatan: hasil cetak 3D (*3D printing*) yang tidak presisi memerlukan proses pengerjaan ulang dan memperlama perakitan komponen; terjadi kerusakan pada konektor M12 yang vital dan memiliki waktu tunggu (*lead time*) yang lama, menyebabkan penundaan signifikan dalam memulai fungsionalitas; dan perakitan kabel yang terlalu rapat (*compact*) di dalam *enclosure* mempersulit proses perakitan dan *troubleshooting* secara keseluruhan. Secara keseluruhan, semua faktor ini berkontribusi besar pada perpanjangan durasi penyelesaian proyek.

3.4 Solusi atas Kendala yang Ditemukan

Berdasarkan kendala yang telah disebutkan pada bagian sebelumnya, penulis menanggulanginya dengan cara berikut:

1) Menyusun rencana pelatihan terfokus pada React dan Tailwind CSS, mungkin melalui kursus daring untuk mempercepat pengembangan UI dan memastikan kualitas *dashboard*.
2) Menambahkan toleransi ukuran desain pada desain 3D, khususnya pada bagian lubang dan mounting sebesar 0,4 mm.

3) Mengimplementasikan kebijakan inventaris untuk menyimpan cadangan komponen kritis (*spare parts*), seperti konektor M12, untuk menghindari *downtime* di masa depan. Selain itu, mencari alternatif supplier lokal yang dapat menyediakan komponen serupa dengan waktu pengiriman yang lebih cepat.
4) Mengadopsi teknik manajemen kabel yang lebih terorganisir, yaitu dengan penggunaan *cable duct* untuk mempermudah perakitan dan pemeliharaan.